# Dynamic Multisource Multipath Routing for Intrusion Tolerance and Lifetime Maximization of Autonomous Wireless Sensor Networks

**Hamid Al-Hamadi and Ing-Ray Chen**

**Department of Computer Science**

**Virginia Tech**

**{hhamadi, irchen}@vt.edu**

*Abstract*— **Multisource multipath data routing to a remote sink node is an effective way to cope with unreliable and malicious nodes in autonomous wireless sensor networks (WSNs). In this paper we analyze the optimal amount of redundancy in terms of the number of source sensors sensing the same physical phenomena and the number of paths through which data are routed to a remote sink node in the presence of unreliable and malicious nodes so that the query success probability is maximized while maximizing the sensor network lifetime. Our dynamic multisource multipath routing algorithm design integrates with a voting-based distributed intrusion detection algorithm to remove malicious nodes from the sensor network. By controlling the redundancy level for multisource multipath and intrusion detection settings dynamically with energy considerations as prescribed by our algorithm, we demonstrate that the lifetime of a query-based autonomous WSN is maximized in response to changing environment conditions including node density, radio range, and node capture rate.**

**Keywords** — ***Wireless sensor networks, multisource multipath routing, intrusion detection, security, reliability, timeliness.***

## I. INTRODUCTION

Advances in wireless sensor networks (WSNs) lead to its wide deployment across many fields. Many WSN applications have high quality of service (QoS) requirements in security, reliability and timeliness. Also many autonomous WSNs are deployed in an unattended manner, so sensor nodes (SNs) are susceptible to capture attacks turning them into malicious inside attackers. SNs have limited resources in energy, computation, transmission range, and storage capability. Thus, the challenge is not only in providing designs satisfying the application specific QoS requirements but also in a way that would consume minimum energy and prolong the lifetime.

Multipath routing is considered an effective way to improve data delivery in WSNs. The basic idea is that the probability of at least one path reaching the sink node or base station increases as we have more paths doing data delivery. While most prior research focused on using multipath routing to improve reliability [10, 22], some attention has been paid to using multipath routing to tolerate insider attacks [16, 18]. Most studies, however, largely ignored the tradeoff between QoS gain vs. energy consumption which can adversely shorten the system lifetime. The tradeoff issue between energy consumption vs. QoS gain becomes much more complicated when inside attackers are present as a path may be broken when a malicious node is on the path. Moreover very likely the system would employ an intrusion detection system (IDS) with the goal to detect and remove malicious nodes. While the literature is abundant in intrusion detection techniques for WSNs [6, 13, 14], the issue of how often intrusion detection should be invoked to remove potentially malicious nodes so that the system lifetime is maximized is largely unexplored. The issue is especially critical for energy-constrained WSNs designed to stay alive autonomously for a long time.

In this paper we address the tradeoff between energy consumption vs. gain in reliability and security with the goal to maximize the lifetime of a query-based autonomous WSN. More specifically, we analyze the optimal amount of redundancy in terms of the number of source SNs sensing the same physical phenomena and the number of paths through which data are routed to a remote sink in the presence of malicious nodes so that the query success probability is maximized while maximizing the WSN lifetime. Our contribution is a model-based analysis methodology by which the optimal multisource multipath redundancy levels and intrusion detection settings are identified for lifetime maximization of query-based WSNs. Untreated in the literature, our multisource multipath routing algorithm design integrates with a voting-based distributed intrusion detection algorithm to remove malicious nodes from the sensor network. By controlling the redundancy level for multisource multipath and intrusion detection settings dynamically with energy considerations as prescribed by our algorithm, we demonstrate that the lifetime of a query-based wireless sensor network is maximized in response to changing environment conditions including node density, radio range, and node capture rate.

## II. RELATED WORK

Over the past few years, numerous protocols have been proposed to detect intrusion in WSNs [6]. In [14], a decentralized rule-based intrusion detection system was proposed by which monitor nodes are responsible for monitoring neighboring nodes using promiscuous listening

and monitoring the collisions for the messages they send to their neighbors. Our host IDS essentially follows this strategy, with the flaws of the host IDS characterized by a false positive probability ($H_{pfp}$) and a false negative probability ($H_{pfn}$). In [14], however, there was no consideration about bad-mouthing attacks by compromised monitor nodes themselves, so if a monitor node is malicious, it can quickly infect others. Our voting-based IDS approach extends from [13] with considerations given to the tradeoff between energy loss vs. security and reliability gain due to employment of voting-based IDS with the goal to prolong the WSN system lifetime.

In the literature, many multipath routing protocols have been proposed for wireless sensor networks. In [18] multiple paths are used to route traffic to the destination using geographic routing, aiming to increase packet delivery ratio in the presence of packet dropping attacks (through blackhole and selective forwarding). A trust based approach is taken by which a sender uses overhearing to monitor if the next nodes forward its packets. Our work differs from theirs in that we concern not only multipath routing, but also energy consumption issues to maximize the WSN system lifetime in the presence of malicious nodes performing bad-mouthing attacks and packet dropping attacks. INSENS [16] is a disjoint multipath routing protocol that aims to tolerate intrusions by using multiple redundant paths to send a message to a destination. It aims to operate correctly in the presence of undetected intruders. However, it relies on the existence of a powerful base station to plan multipath routing, which is normally not available in WSNs, or otherwise would be a single point of failure. Our approach is totally distributed with hop-by-hop formation of multiple paths. SEEM [19] is a multipath routing protocol that also relies on a powerful base station to perform route discovery, maintenance, and route selection. However, it does not consider the existence of malicious nodes and there is no consideration given to detect attacks. Our approach is totally distributed, with considerations given to the presence and detection of malicious nodes in the WSN. In [21], packets are sent over randomized dispersive multipath routes with the aim to avoid black holes resulting from compromised nodes performing packet dropping and/or denial of service attacks. A packet is split into $n$ shares based on coding theory so that if $k$ out of $n$ shares are received then the packet can be reconstructed. The randomized multipath routes generated are dispersive to avoid the black hole and to enhance the probability of at least $k$ out of $n$ shares can reach the receiver. The approach, however, does not consider intrusion detection to detect compromised nodes. Our work considers multipath multisource routing as well to circumvent black hole attacks for intrusion tolerance. In addition, we consider intrusion detection to detect and evict compromised nodes as well as the best rate to invoke intrusion detection to best tradeoff energy consumption vs. security and reliability gain to maximize the system lifetime.

## III. System Model

We consider a WSN with low-power SNs distributed in a geographic area through air-drop. SNs are homogenous with the same initial energy ($E_o$). The deployment area of the WSN is of size $A^2$. SNs are distributed according to a homogeneous spatial Poisson process with intensity λ. We assume the domain is relatively free of obstacles and the WSN is dense enough so that the length of a path connecting two SNs can be approximated by the straight line distance divided by $r$. The transmission power is kept to a minimum such that one-hop radio range ($r$) is used for transmission. Thus, any communication between two nodes with a distance greater than $r$ between them would require a multi-hop. The one-hop radio range can be adjusted to maintain connectivity as the network becomes less dense because of node failures at the expense of more energy consumption.

Environment conditions which could cause a SN to fail with a certain probability include hardware failure ($q$), and transmission failure due to noise and interference ($e$). Moreover, the WSN is vulnerable to sensor captures, i.e., SNs may be captured and compromised. Because of random deployment of SNs (e.g., air drop), we assume all SNs have equal chances of capture with the capture time characterized by a distribution function $F_c(t)$ based on historical data and knowledge about the application environment.

The WSN is cluster-based, where CHs are elected periodically using an energy-saving clustering algorithm (e.g., [17, 23]), and form clusters with non-CH nodes. The clustering algorithm ensures that the energy due to the role of CH is distributed fairly evenly among nodes by performing a fair rotation of the CH role among SNs. Queries can be issued by a mobile user (while moving) and can be issued anywhere in the WSN through a nearby CH. A CH which takes a query to process is called a query processing center (PC). Each query has a strict timeliness requirement ($T_{req}$). The query must be delivered within $T_{req}$ seconds; otherwise, the query fails.

Multisource multipath routing is achieved through two forms of redundancy: (a) source redundancy by which $m_s$ SNs sensing a physical phenomenon in the same feature zone are used to forward sensing data to their CH, referred to as a source CH; (b) path redundancy by which $m_p$ paths are used to relay packets from the source CH to the PC. It has been reported that the number of edge-disjoint paths between nodes is equal to the average node degree with a very high probability [15]. Therefore, when the density is sufficiently high such that the average number of one-hop neighbors is sufficiently larger than $m_p$ and $m_s$, we can effectively result in $m_p$ redundant paths for path redundancy and $m_s$ distinct paths from $m_s$ sensors for source redundancy.

Geographic forwarding is used to route the information between nodes; thus, no path information is maintained. Only the location of the destination SN needs to be known to correctly forward a packet. As part of clustering, a CH knows the locations of SNs within its cluster, and vice versa. We assume that SNs operate in power saving mode (e.g. [7, 20]). Thus, a SN is either active (transmitting or receiving) or in sleep mode. For the transmission and reception energy consumption of sensors, we adopt the energy model in [23].

We assume that the WSN executes a pairwise key establishment protocol in a secure interval after deployment.

Each node establishes pairwise keys with its *k*-hop neighbors, where *k* is large enough to cover a cluster area. Thus, upon electing a new CH, the CH will have pairwise keys with the SNs joining its cluster. Since every SN shares a pairwise key with its CH, a SN can encrypt data sent to the CH for confidentiality and authentication purposes. Due to limited resources, we assume that when a node is compromised, it only performs two most energy conserving attacks, namely, bad-mouthing attacks (recommending a good node as a bad node and a bad node as a good node) when serving as a recommender, and packet dropping attacks when performing packet routing to disrupt the operation of the network.

To detect and remove malicious nodes from the system, a voting-based distributed IDS is applied periodically in every $T_{IDS}$ time interval. How often should $T_{IDS}$ be is a design issue which we aim to identify in this paper. Every node runs a simple *host IDS* using overhearing and promiscuous monitoring techniques (e.g., [5, 14]) to assess its neighbors. The flaws of the host IDS is characterized by a false positive probability ($H_{pfp}$) and a false negative probability ($H_{pfn}$), which are assumed known at deployment time. In each interval, *m* neighbor nodes around a target node will be chosen randomly as voters to decide if the target node is still a good node. The *m* voters share their votes through secure transmission using their pairwise keys. How big should *m* be is another design issue which we aim to identify in this paper. When the majority of voters come to the conclusion that a target node is bad, then the target node is evicted. There is a system-level false positive probability ($P_{fp}$) that the voters can incorrectly identify a good node as a bad node. There is also a system-level false negative probability ($P_{fn}$) that the voters can incorrectly misidentify a bad node as a good node. These two system-level IDS probabilities will be derived based on the *bad-mouthing* attack model in the paper. To provide a unifying metric that considers the above two design tradeoffs, we define the total number of queries the system can answer correctly until it fails as the *lifetime* or the *mean time to failure* (MTTF) of the system which can be translated into the actual system lifetime span based on the query arrival rate.

## IV. PROBABILITY MODEL

In this section we develop a probability model to estimate the MTTF of an autonomous WSN using multisource multipath data forwarding to answer queries issued from a mobile user roaming in the WSN area. The basic idea of our MTTF formulation is that we first deduce the maximum number of queries, $N_q$, the system can possible handle before running into energy exhaustion for the best case in which all queries are processed successfully. Because the system evolves dynamically, the amount of energy spent per query also varies dynamically. Given the query arrival rate $_q$ as input, we can reasonably estimate the amount of energy spent due to query processing and intrusion detection for query *j* based on the query arrival time $t_{Q,j}$. Next we derive the corresponding query success probability $R_q(t_{Q,j})$, that is, the probability that the response to query *j* arriving at time $t_{Q,j}$ is delivered successfully before the query deadline expires. Finally, we compute MTTF as the probability-weighted average of the number of queries the system can handle without experiencing any deadline, transmission, or security failure. More specifically, the MTTF is given by:

$$MTTF = \sum_{i=1}^{N_{q-1}} i\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right) + N_q \prod_{j=1}^{N_q} R_q(t_{Q,j}) \quad (1)$$

Here $\left(\prod_{j=1}^{i} R_q(t_{Q,j})\right)\left(1 - R_q(t_{Q,i+1})\right)$ accounts for the probability of the system being able to successfully execute *i* consecutive queries but failing the $i+1^{\text{th}}$ query. The second term is for the best case in which all queries are processed successfully without experiencing any failure for which the system will have the longest lifetime span.

### A. Network Dynamics

Initially at deployment all SNs are good nodes. Assume that the capture time of a SN follows a distribution function $F_c(t)$ which can be determined based on historical data and knowledge about the target application environment. Then, the probability that a SN is compromised at time *t*, given that it was a good node at time $t\text{-}T_{IDS}$, denoted by $P_c$, is given by:

$$\begin{aligned} P_C &= 1 - P\{X > t \mid X > t - T_{IDS}\} \\ &= 1 - \frac{P\{X > t, X > t - T_{IDS}\}}{P\{X > t - T_{IDS}\}} = 1 - \frac{1 - F_c(t)}{1 - F_c(t - T_{IDS})} \end{aligned} \quad (2)$$

We note that $P_c$ is time dependent. For the special case in which the capture time is exponential distributed with rate $\lambda_c$, $P_c = 1 - e^{-\lambda_c \times T_{IDS}}$. Recall that the voting-based distributed IDS executes periodically with $T_{IDS}$ being the interval. At the $i^{\text{th}}$ IDS execution time (denoted by $t_{I,i}$), a good node may have been compromised with probability $P_c$ since the previous IDS execution time $(t_{I,i-1})$. Let $n_{good}(t)$ and $n_{bad}(t)$ denote the numbers of good and bad neighbor nodes at time *t,* respectively, with $n_{good}(t) + n_{bad}(t) = n(t)$. Then, the population of good and bad neighbor nodes at time $t_{I,i}$ just prior to IDS execution can be recursively estimated from the population of good and bad neighbor nodes at time $t_{I,i-1}$ :

$$\begin{aligned} n_{good}(t_{I,i}) &= n_{good}(t_{I,i-1}) - n_{good}(t_{I,i-1}) \times P_c \\ n_{bad}(t_{I,i}) &= n_{bad}(t_{I,i-1}) + n_{good}(t_{I,i-1}) \times P_c \end{aligned} \quad (3)$$

With $n_{good}(t)$ and $n_{bad}(t)$ in hand, the system-level false positive probability ($P_{fp}$) and false negative probability ($P_{fn}$) as a resulting of executing voting-based IDS are as follows:

$$\begin{aligned} P_{fp} \text{ or } P_{fn} \equiv & \sum_{i=0}^{m-m_{maj}} \left[ \frac{C\binom{n_{bad}}{m_{maj}+i} \times C\binom{n_{good}}{m-(m_{maj}+i)}}{C\binom{n_{bad}+n_{good}}{m}} \right] \\ + & \sum_{i=0}^{m-m_{maj}} \left[ \frac{C\binom{n_{bad}}{i} \times \sum_{j=m_{maj}-i}^{m-i} \left[ C\binom{n_{good}}{j} \times \omega^j \times C\binom{n_{good-j}}{m-i-j} \times (1-\omega)^{m-i-j} \right]}{C\binom{n_{bad}+n_{good}}{m}} \right] \end{aligned} \quad (4)$$

where $m_{maj}$ is the minimum majority of *m*, e.g., 3 is the minimum majority of 5, and $\omega$ is $H_{pfp}$ for calculating $P_{fp}$ and $H_{pfn}$ for calculating $P_{fn}$. We explain Equation 4 for the false positive probability at time *t* below. The explanation to the false negative probability is similar. A false positive results when the majority of the voters vote against the target node

(which is a good node) as compromised. The first term in Equation 4 accounts for the case in which more than 1/2 of the voters selected from the target node's neighbors are bad sensors who, as a result of performing bad-mouthing attacks, will always vote a good node as a bad node to break the functionality of the WSN. Here the denominator is the total number of combinations to select *m* voters out of all neighbor nodes, and the numerator is the total number of combinations to select at least $m_{maj}$ bad voters out of $n_{bad}$ nodes and the remaining good voters out of $n_{good}$ nodes. The second term accounts for the case in which more than 1/2 of the voters selected from the neighbors are good nodes but unfortunately some of these good nodes mistakenly misidentify the target nodes as a bad node with probability $H_{pfp}$, resulting in more than 1/2 of the voters (some of those may be bad nodes) voting against the target node. Here the denominator is again the total number of combinations to select *m* voters out of all neighbor nodes, and the numerator is the total number of combinations to select *i* bad voters not exceeding the majority $m_{maj}$, *j* good voters who diagnose incorrectly with $i + j \geq m_{maj}$, and the remaining $m - i - j$ good voters who diagnose correctly.

After the voting-based IDS is executed, some good nodes will be misidentified as bad nodes with probability $P_{fp}$ and will be mistakenly removed from the WSN. Consequently, we need to adjust the population of good nodes after IDS execution. Let $\overline{n_{good}(t)}$ be the number of good neighbor nodes at time *t* right after IDS execution. Then,

$$\overline{n_{good}(t_{I,t})} = n_{good}(t_{I,i}) - n_{good}(t_{I,i}) \times P_{fp} \quad (5)$$

On the other hand, some bad nodes will remain in the system because the voting-based IDS fails to identify them with probability $P_{fn}$. Let $\overline{n_{bad}(t)}$ be the number of bad neighbor nodes at time *t* right after IDS execution. Then,

$$\overline{n_{bad}(t_{I,t})} = n_{bad}(t_{I,i}) - n_{bad}(t_{I,i}) \times (1 - P_{fn}) \quad (6)$$

As the capture attack is totally random, the probability that any neighbor node is a bad node at time *t*, denoted by $Q_{c,j}(t)$, thus is given by:

$$Q_{c,j}(t_{I,i}) = \frac{\overline{n_{bad}(t_{I,t})}}{\overline{n_{bad}(t_{I,t})} + \overline{n_{good}(t_{I,t})}} \quad (7)$$

$Q_{c,j}(t)$ derived above provides critical information as a bad node can perform packet dropping attacks if it is on a path from source SNs to the PC. Here we note that the node population density is evolving because of some nodes being compromised and some being detected and evicted by the IDS dynamically. The node population remains the same until the next IDS execution (after $T_{IDS}$ seconds) because the IDS only detects and evicts nodes periodically (as typically node hardware/software failure happens less frequently than security failure). Denote the node population density at time *t* by $\lambda(t)$ with $\lambda(0) = \lambda$. Then, $\lambda(t)$ can be computed by:

$$n(t_{I,i}) = \overline{n_{bad}(t_{I,t})} + \overline{n_{good}(t_{I,t})} \quad (8)$$

$$\lambda(t_{I,i}) = \frac{n(t_{I,i})}{\pi r^2} \quad (9)$$

### B. Query Success Probability

There are three ways by which data forwarding from $SN_j$ to $SN_k$ could fail: (a) transmission speed violation; (b) sensor/channel failures; and (c) $SN_j$ is compromised.

The first source of failure, transmission speed violation, accounts for query deadline violation. To know the failure probability due to transmission speed violation, we first derive the minimum hop-by-hop transmission speed required to satisfy the query deadline $T_{req}$. Let $d_{SN\text{-}CH}$ be the *expected* distance between a SN (selected to report sensor readings) and its CH and $d_{CH\text{-}PC}$ be the *expected* distance between the source CH and the PC accepting the query result. Given a query deadline $T_{req}$ as input, a data packet from a SN through its CH to the PC must reach the PC within $T_{req}$. Thus, the minimum hop-by-hop transmission speed denoted by $S_{req}$ is given by:

$$S_{req} = \frac{d_{SN-CH} + d_{CH-PC}}{T_{req}} \quad (10)$$

Since a SN becomes a CH with probability *p* and all the sensors are distributed in the area in accordance with a spatial Poisson process with intensity λ, CHs and non-CH SNs will also be distributed in accordance with a spatial Poisson process with rates $p\lambda$ and $(1\text{-}p)\lambda$ respectively. Non-CH SNs thus would join the closest CH to form a Voronoi cell [1] corresponding to a cluster in the WSN. It can be shown that the average number of non-CH SNs in each Voronoi cell is $(1\text{-}p)/p$ and the expected distance from a SN to its CH is given by $d_{SN-CH} = 1/2(p\lambda)^{1/2}$. On the other hand, since a query may be issued from anywhere by the mobile user to a CH (which serves as the PC) and the source CH requested by the query also can be anywhere in the WSN, $d_{CH\text{-}PC}$ essentially is the average distance between any two CHs in the WSN. Given location randomness of CHs in the square area $A^2$, it can be shown geometrically that the average distance between any two CHs is $d_{CH\text{-}PC}$=0.382*A*. With the knowledge of $d_{SN\text{-}CH}$ and $d_{CH\text{-}PC}$, we can also estimate the average numbers of hops to forward data from a SN to the source CH, denoted by $N_{SC}^h$, and the average numbers of hops to forward data from the source CH to the PC, denoted by $N_{CP}^h$, by $N_{SC}^h = d_{SN-CH}/r$ and $N_{cp}^h = d_{CH-PC}/r$ where *r* is radio range.

Let $Q_{t,jk}$ denote the probability that the forwarding speed from $SN_j$ to $SN_k$ would violate the minimum speed requirement, thus leading to a query deadline violation failure. To calculate $Q_{t,jk}$ we need to know the transmission speed $S_{jk}$ from $SN_j$ to $SN_k$. This can be dynamically measured by $SN_j$. If $S_{jk}$ is above $S_{req}$ then $Q_{t,jk} = 0$; otherwise, $Q_{t,jk} = 1$. In general $S_{jk}$ is not known until runtime. If $S_{jk}$ is uniformly distributed within a range $[a, b]$, then $Q_{t,jk}$ can be computed as:

$$Q_{t,jk} = cdf(S_{jk} \leq S_{req}) = \frac{S_{req} - a}{b - a} \quad (11)$$

The second source of failure is due to sensor failure or channel failure. Let $Q_{r,j}$ denote the probability of failure due to sensor failure or channel failure. Since *q* is the hardware failure probability and $e_j$ is transmission failure probability of node *j*, given as input, $Q_{r,j}$ can be estimated by:

$$Q_{r,j} = 1 - [(1-q)(1-e_j)] \quad (12)$$

The third source of failure is due to node *j* being compromised and thus the packet is dropped. We make use of $Q_{c,j}(t)$ derived earlier in Equation 7. By combining these three

failure probabilities we obtain $Q_{rtc,jk}$, the probability of $SN_j$ failing to relay a data packet to a one-hop neighbor $SN_k$ because of either speed violation, sensor/channel failure, or $SN_j$ being compromised, as:

$$Q_{rtc,jk} = 1 - [(1 - Q_{r,j})(1 - Q_{t,jk})(1 - Q_{c,j})] \quad (13)$$

By using this one-hop failure probability, we next compute the success probability for $SN_j$ to transmit a packet to at least one next-hop SN neighbor along the direction of the destination node as:

$$\theta_j = 1 - \prod_{k=1}^{f \times n} Q_{rtc,jk} \quad (14)$$

where $f$ =1/4 to account for the fact that only neighbor SNs in the quadrant toward the destination node can perform geographic forwarding; $n$ is the number of neighbor SNs of node $j$ as given in Equation 8.

Since on average there will be $N_{CP}^h$ hops on a path from the source CH to the PC, a data packet transmitted along the path is successfully delivered only if it is delivered successful hop-by-hop without experiencing any speed violation failure, hardware/channel failure, or packet dropping failure, for $N_{CP}^h$ hops. Consequently, the probability of a single path between the source CH and the PC being able to deliver data successfully is given by:

$$\Theta(N_{CP}^h) = (\prod_{j=1}^{N_{CP}^h - 1} \theta_j) \times (1 - Q_{rtc,\ N_{CP}^h (N_{CP}^h + 1)}) \quad (15)$$

For path redundancy, we create $m_p$ paths between the source CH and the PC. The $m_p$ paths are formed by choosing $m_p$ SNs in the first hop and then choosing only one SN in each of the subsequent hops. The source CH will fail to deliver data to the PC if one of the following happens: (a) none of the SNs in the first hop receives the message; (b) in the first hop, $i$ ($1 \leq i < m_p$) SNs receive the message, and each of them attempts to form a path for data delivery; however, all $i$ paths fail to deliver the message because the subsequent hops fail to receive the broadcast message; (c) in the first hop, at least $m_p$ SNs receive the message from the source CH from which $m_p$ SNs are randomly selected to forward data, but all $m_p$ paths fail to deliver the message because the subsequent hops fail to receive the message. Summarizing above, the probability of the source CH failing to deliver data to the PC is given by:

$$\begin{aligned} Q_{fp}^{m_p} &= 1 - \theta_1 + \\ &\sum_{|I| < m_p} \{[\prod_{i \in I} (1 - Q_{rtc,1i})](\prod_{i \notin I} Q_{rtc,1i})\} \{\prod_{i \in I} [1 - \Theta_i(N_{CP}^h - 1)]\} + \\ &\sum_{|I| \geq m_p} \{[\prod_{i \in I} (1 - Q_{rtc,1i})](\prod_{i \notin I} Q_{rtc,1i})\} \{\prod_{\substack{i \in M, \\ M \subseteq I, \\ |M| = m_p}} [1 - \Theta_i(N_{CP}^h - 1)]\} \end{aligned} \quad (16)$$

Following the same derivation to Equation 15, the success probability of a single path from a SN to its CH is given by:

$$\Theta(N_{SC}^h) = (\prod_{j=1}^{N_{SC}^h - 1} \theta_j) \times (1 - Q_{rt,\ N_{SC}^h (N_{SC}^h + 1)}) \quad (17)$$

For source redundancy we use $m_s$ SNs to report query responses to their source CH. The probability that all $m_s$ SNs fail to deliver data to their CH is given by:

$$Q_{fs}^{m_s} = \prod_{i=1}^{m_s} [1 - \Theta_i(N_{SC}^h)] \quad (18)$$

Consequently, the failure probability of data delivery from $m_s$ SNs to the CH, and subsequently using $m_p$ paths to relay data from CH to PC, is given by:

$$Q_f = 1 - (1 - Q_{fp}^{m_p})(1 - Q_{fs}^{m_s}) \quad (19)$$

Therefore, the query success probability is given by:

$$R_q = 1 - Q_f \quad (20)$$

Note that in the above derivation we omit time for brevity. More precisely, $R_q$ derived above should be $R_q(t_{Q,i})$ since the query success probability is a function of time, depending on the node count (Equation 8) and population density (Equation 9) at the $i^{th}$ query's execution time (i.e., at time $t_{Q,i}$).

### C. Energy Consumption

Due to space limit, we sketch the procedure for computing $N_q$, the maximum number of queries the system can possible handle before running into energy exhaustion, required by Equation 1. The basic idea is to estimate the amount of energy consumed for query processing, intrusion detection, and clustering, respectively, based on an energy model [23]. Then, we can estimate $N_q$ by the fact that the total energy consumed due to intrusion detection, clustering and query processing is equal to the system energy.

## V. Performance Evaluation

In this section, we present numerical results obtained from the evaluation of our probability model given in Section IV. Without loss of generality, we consider an example WSN consisting of 1500 nodes deployed in a square area of $A^2$ (400m×400m). Nodes are distributed in the area following a Poisson process with density $\lambda$ =15 nodes/(40×40 $m^2$) at deployment time. The radio range $r$ is 40m. So initially a SN has n=$\lambda \times \pi r^2$=15 neighbor SNs. The probability of a SN becoming a CH is $p$=1%. So initially a cluster has 1/$p$=100 nodes and there are 15 clusters in the system. Each SN has an initial energy level $E_o$ = 10 Joules. The energy parameters used by the radio module are adopted from [17, 23] . The energy dissipation $E_{elec}$ to run the transmitter and receiver circuitry is 50 nJ/bit. The energy used by the transmit amplifier to achieve an acceptable signal to noise ratio ($\varepsilon_{amp}$) is 10 pJ/bit/$m^2$. The query arrival rate $\lambda_q$ is a variable and is set to 1 query/sec to reveal points of interest. The query deadline $T_{req}$ is strict and set to between 0.3 and 1 sec. The inter-arrival time in between captures ($T_{comp}$) is between 4 and 28 days, corresponding to a capture rate ($\lambda_c$) of once per 4 days to once per 28 days. The host IDS false positive probability and false negative probability ($H_{pfp}$ and $H_{pfn}$) vary between 1% and 5% to reflect

the host intrusion detection strength as in [14]. Our objective is to identify the best setting in terms of $m_p$ (path redundancy), $m_s$ (source redundancy), $m$ (the number of voters for intrusion detection) and $T_{IDS}$ (the intrusion detection interval) to maximize MTTF.

In Fig. 1, we show MTTF vs. ($m_p$, $m_s$) for three cases: (a) there are no malicious nodes and no intrusion detection (the top curve); (b) there are malicious nodes but there is no intrusion detection (the bottom curve); (c) there are malicious nodes and there is intrusion detection (the middle two curves). In each case we observe the existence of an optimal ($m_p$, $m_s$) value under which MTTF is maximized. When there are no malicious nodes (the top curve), the optimal ($m_p$, $m_s$) is (3,3). When there are malicious nodes, and no intrusion detection is used, the optimal ($m_p$, $m_s$) value becomes (7,7) because using higher redundancy in multisource multipath routing is necessary to cope with malicious nodes. When intrusion detection is used (middle curves), there exists an optimal $m$ value to maximize MTTF. In Fig. 1, $m$=5 yields a higher MTTF value than $m$=3 because in this scenario the attack rate is relatively high (once a week), so a higher number of voters is needed to cope with and detect bad nodes more effectively. We observe that the maximum MTTF is sensitive to $T_{comp}$ and $m$. Table 1 below summarizes the effect of $T_{comp}$ and $m$ on optimal ($m_p$, $m_s$) values under which MTTF is maximized. As the number of voters in intrusion detection ($m$) increases, the optimal ($m_p$, $m_s$) redundancy level decreases. This is because increasing $m$ has the effect of detecting and evicting bad nodes more effectively, thus requiring a lower level of redundancy in ($m_p$, $m_s$) to cope with packet dropping attacks by bad nodes.

On the other hand, when given a $T_{comp}$ there exists an optimal $m$ value that will maximize MTTF. Table 2 summarizes the effect of $T_{comp}$ on the optimal $m$ value at which MTTF is maximized. When the node capture rate increases from once per 3 weeks ($T_{comp}$ = 3 weeks) to once a week ($T_{comp}$ = 1 week), the optimal $m$ value goes from 3 to 7. The reason is that as the capture rate increases, there are more and more malicious nodes in the system, so using more voters (e.g. $m$ =7) can help identify and evict malicious nodes, thus increasing the query success probability and consequently increasing the MTTF value. Again the system is better off this way to cope with increasing malicious node population for lifetime maximization even though more energy is consumed due to more voters being used.

Lastly $T_{IDS}$ is also a tunable parameter to maximize MTTF. Fig. 2 shows MTTF vs. $T_{IDS}$ with varying $T_{comp}$ values. It exhibits the trend that as the capture rate increases (a smaller $T_{comp}$ value), the optimal $T_{IDS}$ at which MTTF is maximized must decrease to cope with malicious attacks. Furthermore, the optimal $T_{IDS}$ value increases as $m$ increases. The reason is that as the number of voters ($m$) increases so the intrusion detection capability increases per invocation, there is no need to invoke intrusion detection too often so as not to waste energy and adversely shorten the system lifetime. Table 3 summarizes the effect of $T_{comp}$ and $m$ on the optimal $T_{IDS}$ value at which MTTF is maximized.

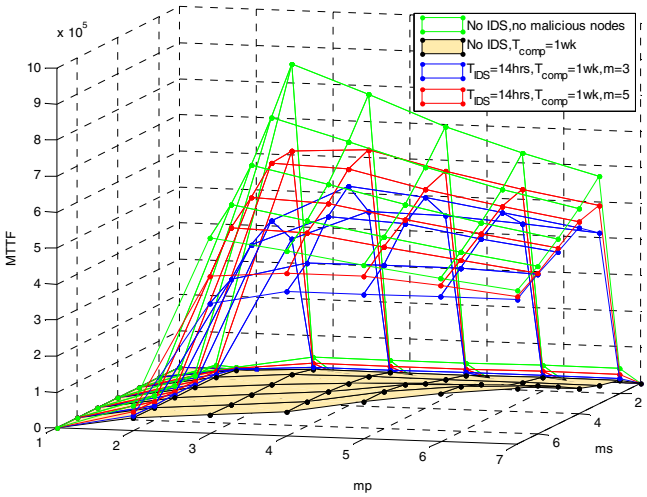


Figure 1. MTTF vs. ($m_p$, $m_s$): (3, 4) and (4, 4) are Optimal When $m$=5 and 3, Respectively.

TABLE 1. OPTIMAL ($m_p$, $m_s$) WITH VARYING $T_{comp}$ AND $m$.

| | $m$=3 | 5 | 7 |
|---|---|---|---|
| $T_{comp}$=4 days | ($m_p$, $m_s$) =(5,7) | (4,6) | (4,5) |
| 1 week | (4,4) | (3,4) | (3,3) |
| 3weeks | (3,3) | (3,3) | (3,3) |

TABLE 2. OPTIMAL $m$ WITH VARYING $T_{comp}$ AND $T_{IDS}$.

| | $T_{IDS}$=1hr | 4hrs | 14hrs | 20hrs | 24hrs |
|---|---|---|---|---|---|
| $T_{comp}$=4 days | $m$=5 | 7 | 7 | 7 | 7 |
| 1 week | 5 | 5 | 7 | 7 | 7 |
| 2 weeks | 5 | 5 | 5 | 5 | 7 |
| 3 weeks | 3 | 3 | 3 | 5 | 5 |

TABLE 3. OPTIMAL $T_{IDS}$ WITH VARYING $T_{comp}$ AND $m$.

| | $m$ =3 | 5 | 7 |
|---|---|---|---|
| $T_{comp}$=4 days | $T_{IDS}$=6 hours | 6 | 10 |
| 1 week | 10 | 10 | 14 |
| 2 weeks | 12 | 20 | 28 |
| 3 weeks | 16 | 28 | 44 |

Tables 1, 2 and 3 presented above are numerical solutions generated from evaluating the analytical equations derived in Section IV, given node density λ, radio range $r$, and node capture rate $\lambda_c$ as input. As the system evolves, all these input parameter values may change, that is, λ will decrease as described by Equation 9, radio range $r$ will increase to maintain connectivity as more nodes fail or are evicted from the system, and $\lambda_c$ may evolve depending on the instantaneous attacker strength. Lookup tables such as Tables 1, 2 and 3 are

built at static time, covering a wide range of ($\lambda$, $r$, $\lambda_c$) values as input. Our dynamic multisource multipath routing algorithm then utilizes these lookup tables built at static time to perform a simple lookup operation to decide the optimal settings of ($m_p$, $m_s$, $m$, $T_{IDS}$) to maximize MTTF at runtime.

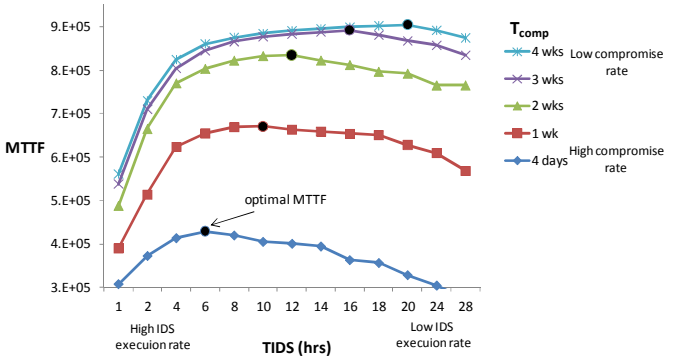


Figure 2. Effect of Capture Rate on MTTF.

## VI. CONCLUSION

In this paper we provided a solution to the issue of dynamic and adaptive multisource multipath routing for intrusion tolerance and lifetime maximization in autonomous wireless sensor networks. We developed a novel probability model to analyze the best multisource multipath redundancy level in terms of path redundancy ($m_p$) and source redundancy ($m_s$), as well as the best intrusion detection settings in terms of the number of voters ($m$) and the intrusion invocation interval ( $T_{IDS}$) under which the lifetime of a query-based wireless sensor network may be maximized in the presence of unreliable wireless communication and malicious nodes. Our dynamic multisource multipath routing algorithm utilizes the analysis result to determine the optimal system settings for redundancy and intrusion detection based on the sensed environmental conditions at runtime, thus resulting in the system achieving its maximum lifetime.

In the future we plan to consider more sophisticated attacker models, e.g., a smart adversary that can perform more targeted attacks, capture certain strategic nodes with higher probability, alternate between benign and malicious behavior and collude with other attackers to avoid intrusion detection. This paper addresses the best redundancy level for multisource multipath routing, i.e., how many sources and how many paths one should use for multisource multipath routing to maximize the system lifetime. In the future, we plan to explore trust management [2, 3, 8] for performing intrusion detection augmented with fuzzy failure criteria [4] to address the issue of what paths one should use to avoid untrustworthy, malicious nodes to further enhance WSN survivability. This may involve the use of trust-based admission control strategies [9, 11, 12] to increase the probability of path success probability for data delivery.

## REFERENCES


[1] S. Bandyopadhyay and E. J. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks " *22nd Conf. of IEEE Computer and Communications*, pp. 1713-1723, 2003.

[2] F. Bao, I. R. Chen, M. Chang, and J. Cho, "Hierarchical Trust Management for Wireless Sensor Networks and its Applications to Trust-Based Routing and Intrusion Detection," *IEEE Trans. Netw. Service Manag.,* vol. 9, no. 2, pp. 161-183, 2012.

[3] F. Bao, I. R. Chen, M. Chang, and J. H. Cho, "Trust-Based Intrusion Detection in Wireless Sensor Networks," *IEEE Int. Conf. on Communications*, Kyoto, Japan, June 2011.

[4] F. B. Bastani, I. R. Chen, and T. Tsao, "Reliability of Systems with Fuzzy-Failure Criterion," *Annu. Reliability and Maintainability Symp.*, 1994.

[5] V. Bhuse and A. Gupta, "Anomaly intrusion detection in wireless sensor networks," *J. High Speed Netw.,* vol. 15, no. 1, pp. 33-51, 2006.

[6] S. Bo, L. Osborne, X. Yang, and S. Guizani, "Intrusion detection techniques in mobile ad hoc and wireless sensor networks," *IEEE Wireless Commun.,* vol. 14, no. 5, pp. 56-63, 2007.

[7] G. Bravos and A. G. Kanatas, "Energy consumption and trade-offs on wireless sensor networks," *16th IEEE Int. Symp. on Personal, Indoor and Mobile Radio Communications*, pp. 1279-1283, 2005.

[8] I. R. Chen, F. Bao, M. Chang, and J. H. Cho, "Trust Management for Encounter-based Routing in Delay Tolerant Networks," *IEEE Global Communications Conf.*, Miami, Florida, USA, Dec. pp. 1-6, 2010.

[9] I. R. Chen and T. H. Hsi, "Performance analysis of admission control algorithms based on reward optimization for real-time multimedia servers," *Performance Evaluation,* vol. 33, no. 2, pp. 89-112, 1998.

[10] I. R. Chen, A. P. Speer, and M. Eltoweissy, "Adaptive Fault-Tolerant QoS Control Algorithms for Maximizing System Lifetime of Query-Based Wireless Sensor Networks," *IEEE Trans. on Dependable and Secure Computing,* vol. 8, no. 2, pp. 161-176, 2011.

[11] S. T. Cheng, C. M. Chen, and I. R. Chen, "Dynamic quota-based admission control with sub-rating in multimedia servers," *Multimedia systems,* vol. 8, no. 2, pp. 83-91, 2000.

[12] S. T. Cheng, C. M. Chen, and I. R. Chen, "Performance evaluation of an admission control algorithm: dynamic threshold with negotiation," *Performance Evaluation,* vol. 52, no. 1, pp. 1-13, 2003.

[13] J. H. Cho, I. R. Chen, and P. G. Feng, "Effect of Intrusion Detection on Reliability of Mission-Oriented Mobile Group Systems in Mobile Ad Hoc Networks," *IEEE Trans. Rel.,* vol. 59, no. 1, pp. 231-241, 2010.

[14] A. P. R. da Silva, M. H. T. Martins, B. P. S. Rocha, A. A. F. Loureiro, L. B. Ruiz, and H. C. Wong, "Decentralized intrusion detection in wireless sensor networks," *1st ACM Workshop on Quality of Service & Security in Wireless and Mobile Networks*, Montreal, Quebec, Canada, 2005.

[15] B. Deb, S. Bhatnagar, and B. Nath, "ReInForM: reliable information forwarding using multiple paths in sensor networks," *28th IEEE Local Computer Networks*, Bonn, Germany, pp. 406-415, 2003.

[16] J. Deng, R. Han, and S. Mishra, "INSENS: Intrusion-tolerant routing for wireless sensor networks," *Computer Communications,* vol. 29, no. 2, pp. 216-230, 2006.

[17] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.,* vol. 1, no. 4, pp. 660-670, 2002.

[18] K. D. Kang, K. Liu, and N. Abu-Ghazaleh, "Securing Geographic Routing in Wireless Sensor Networks," *9th Annu. Cyber Security Conf. on Information Assurance*, Albany, NY, USA, 2006.

[19] N. Nasser and Y. Chen, "SEEM: Secure and energy-efficient multipath routing protocol for wireless sensor networks," *Computer Communications,* vol. 30, no. 11-12, pp. 2401-2412, 2007.

[20] S. Qun, "Power Management in Networked Sensor Radios A Network Energy Model," *IEEE Sensors Applications Symp.*, pp. 1-5, 2007.

[21] T. Shu, M. Krunz, and S. Liu, "Secure Data Collection in Wireless Sensor Networks Using Randomized Dispersive Routes," *IEEE Trans. Mobile Comput.,* vol. 9, no. 7, pp. 941-954, 2010.

[22] Y. Yang, C. Zhong, Y. Sun, and J. Yang, "Network coding based reliable disjoint and braided multipath routing for sensor networks," *J. Netw. Comput. Appl.,* vol. 33, no. 4, pp. 422-432, 2010.

[23] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mobile Comput.,* vol. 3, no. 4, pp. 366-379, 2004.